Report on Forest Fire Area Prediction using SVR

---

1. Introduction

Forest fires are a major environmental issue, causing significant damage to ecosystems, property, and human life. Predicting the burnt area of a forest fire is important for resource planning and disaster management.
This project explores the use of **Support Vector Regression (SVR)** to predict the burnt area of forest fires using the publicly available **forestfires.csv** dataset.

---

2. Dataset Description

The dataset forestfires.csv contains meteorological and environmental attributes that influence forest fires.

Key Features:

- **X, Y:** Spatial coordinates within the forest map.
- **month, day:** Temporal attributes of the fire occurrence.
- **FFMC, DMC, DC, ISI:** Fire weather indices.
- **temp, RH, wind, rain:** Weather conditions (temperature, relative humidity, wind speed, rainfall).
- **area:** Target variable – burnt area of the forest (in hectares).

The dataset is relatively small but contains useful predictors for understanding fire behavior.

```python
import pandas as pd

df = pd.read_csv('forestfires.csv')
display(df.head())
```

	X	Y	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area
0	7	5	mar	fri	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	0.0
1	7	4	oct	tue	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	0.0
2	7	4	oct	sat	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	0.0
3	8	6	mar	fri	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	0.0
4	8	6	mar	sun	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	0.0

---

3. Data Loading and Preprocessing

The dataset was loaded into a **pandas DataFrame** for analysis.

Steps:

1. **Missing Values:** Checked for missing values – none were found.
2. **Categorical Encoding:** Categorical variables (month, day) were **one-hot encoded**.
3. **Feature Scaling:** All numerical features were standardized using **StandardScaler** to ensure equal contribution during model training.

This preprocessing ensured the data was clean and ready for machine learning.

```python
display(df.isnull().sum())

categorical_cols = df.select_dtypes(include=['object']).columns
numerical_cols = df.select_dtypes(include=['int64', 'float64']).columns

df_encoded = pd.get_dummies(df, columns=categorical_cols, drop_first=True)

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
df_encoded[numerical_cols] = scaler.fit_transform(df_encoded[numerical_cols])

display(df_encoded.head())
```

	0
X	0
Y	0
month	0
day	0
FFMC	0
DMC	0
DC	0
ISI	0
temp	0
RH	0
wind	0
rain	0

dtype: int64

	X	Y	FFMC	DMC	DC	ISI	temp	RH	wind	rain	...	month_may	month_nov	month_oct	month_sep	day_mon	day_sat	day_sun	day_thu	day_
0	1.008313	0.569860	-0.805959	-1.323326	-1.830477	-0.860946	-1.842640	0.411724	1.498614	-0.073268	...	False	False	False	False	False	False	False	False	Fa
1	1.008313	-0.244001	-0.008102	-1.179541	0.488891	-0.509688	-0.153278	-0.692456	-1.741756	-0.073268	...	False	False	True	False	False	False	False	False	T
2	1.008313	-0.244001	-0.008102	-1.049822	0.560715	-0.509688	-0.739383	-0.692456	-1.518282	-0.073268	...	False	False	True	False	False	True	False	False	Fa
3	1.440925	1.383722	0.191362	-1.212361	-1.898266	-0.004756	-1.825402	3.233519	-0.009834	0.603155	...	False	False	False	False	False	False	False	False	Fa
4	1.440925	1.383722	-0.243833	-0.931043	-1.798600	0.126966	-1.291012	3.356206	-1.238940	-0.073268	...	False	False	False	False	False	False	True	False	Fa

5 rows × 28 columns

4. Data Splitting

- The dataset was divided into **Training (80%)** and **Testing (20%)** subsets.
- Splitting ensured that model evaluation was done on unseen data for fair performance assessment.

```
from sklearn.model_selection import train_test_split

X = df_encoded.drop('area', axis=1)
y = df_encoded['area']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

---

5. Model Building and Training

Initially, an attempt was made to use **Support Vector Classification (SVC)**. However, since the target variable area is **continuous**, classification was not suitable.

Therefore:

- A **Support Vector Regressor (SVR)** was selected.
- The SVR model was trained on the **training set** using default parameters.

```
from sklearn.svm import SVR

svm_model = SVR()
svm_model.fit(X_train, y_train)
```

```
SVR
SVR()
```

---

6. Model Evaluation

The trained model was evaluated on the **test set** using two standard regression metrics:

- **Mean Squared Error (MSE):** 2.95
- **R-squared (R^2):** -0.013

Interpretation:

- The **low R^2 score (negative)** indicates that the model performs **worse than a baseline mean predictor**.

- Predictions were not reliable and failed to capture the variability of burnt area effectively.

```
from sklearn.metrics import mean_squared_error, r2_score

y_pred = svm_model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error (MSE): {mse}')
print(f'R-squared (R2): {r2}')
```

```
Mean Squared Error (MSE): 2.9544514724505935
R-squared (R2): -0.013633299733563309
```

---

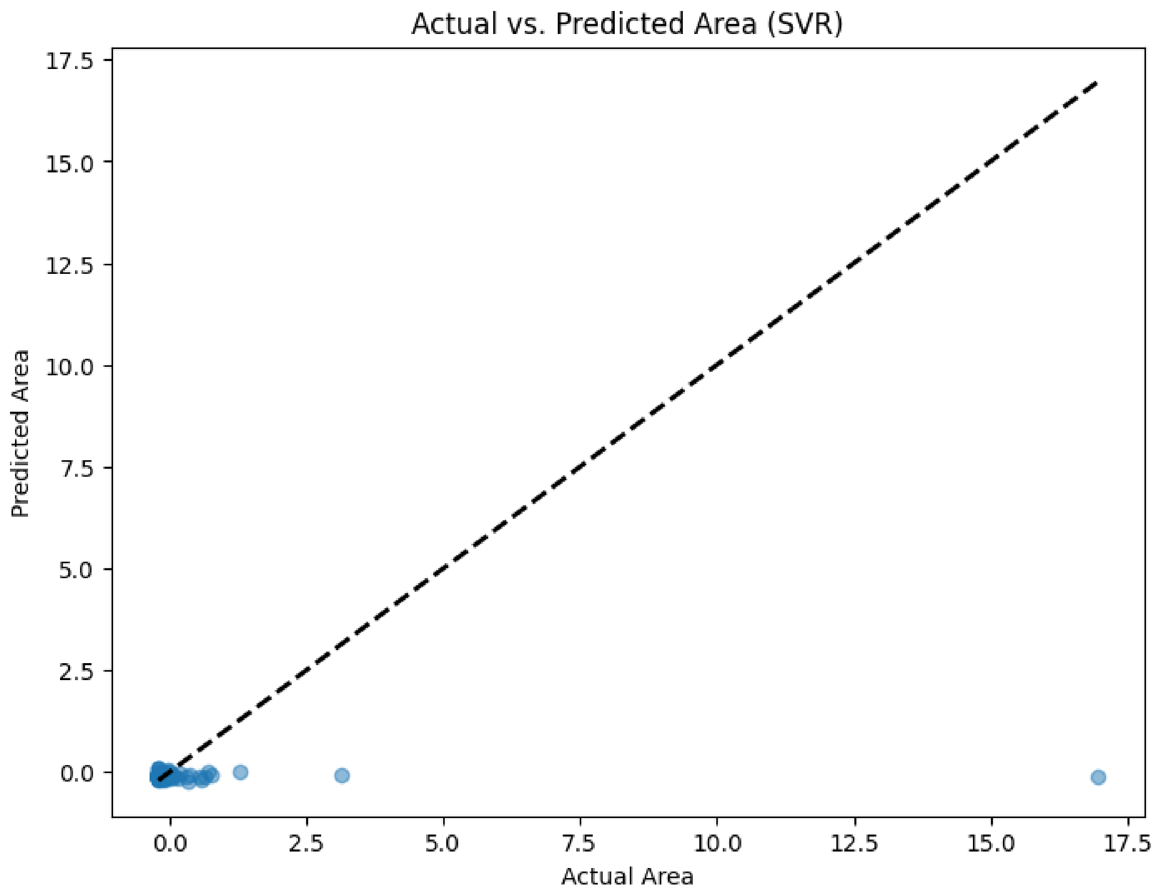
7. Visualization

To visualize performance, a **scatter plot** of actual vs. predicted burnt area was generated.

- The plot revealed that predicted values were **poorly aligned** with actual values.
- This confirmed the **weak predictive power** of the current SVR model.

```
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, alpha=0.5)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2)
plt.xlabel("Actual Area")
plt.ylabel("Predicted Area")
plt.title("Actual vs. Predicted Area (SVR)")
plt.show()
```



---

8. Conclusion

The SVR model was implemented successfully but yielded **poor predictive performance**:

- **MSE ≈ 2.95**
- **R^2 ≈ -0.01**

This suggests that the current SVR configuration is **not effective** for predicting burnt area in the given dataset.

---

9. Recommendations & Next Steps

To improve performance, the following strategies are suggested:

1. **Hyperparameter Tuning**
    - Experiment with SVR kernels (linear, poly, rbf)
    - Adjust parameters (C, epsilon, gamma)
2. **Feature Engineering**

- Apply log-transform to the skewed target variable (area)
- Create interaction features (e.g., temp × wind, rain × humidity)

3. **Alternative Algorithms**
    - **Tree-based models** (Random Forest, Gradient Boosting)
    - **Ensemble methods**
    - **Neural Networks** for capturing complex relationships

---

10. References

- Cortez, Paulo, and Aníbal Morais. "A Data Mining Approach to Predict Forest Fires using Meteorological Data." *Proceedings of the 13th Portuguese Conference on Artificial Intelligence* (2007).
- Scikit-learn Documentation: https://scikit-learn.org